Function Calls & Stack examples

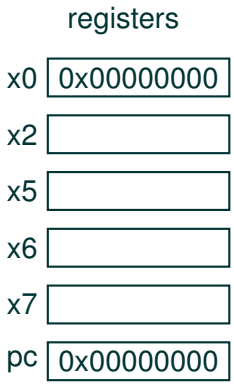
Stefan Mangard

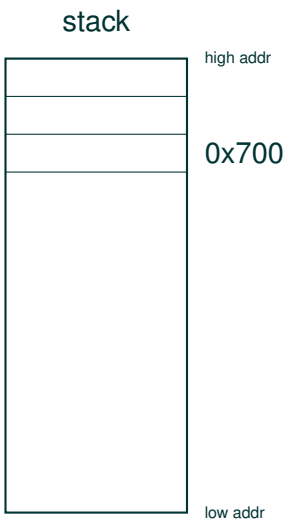
Computer Organization and Networks
Graz University of Technology

Push and Pop

Push and pop

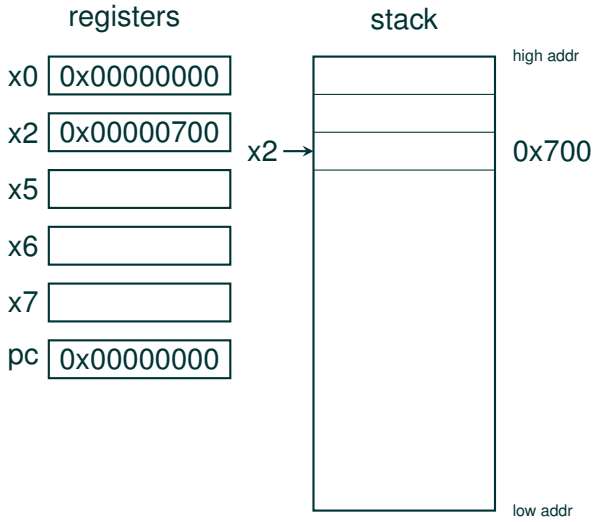
```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```





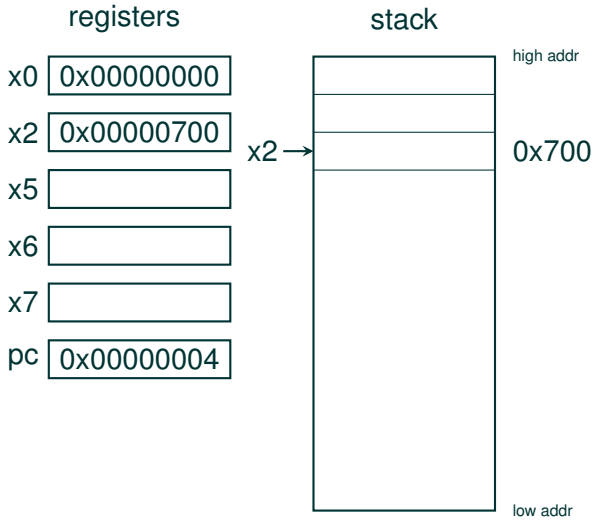
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



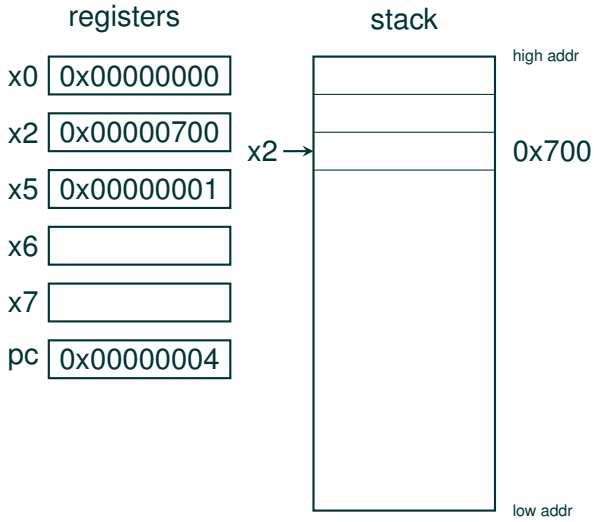
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



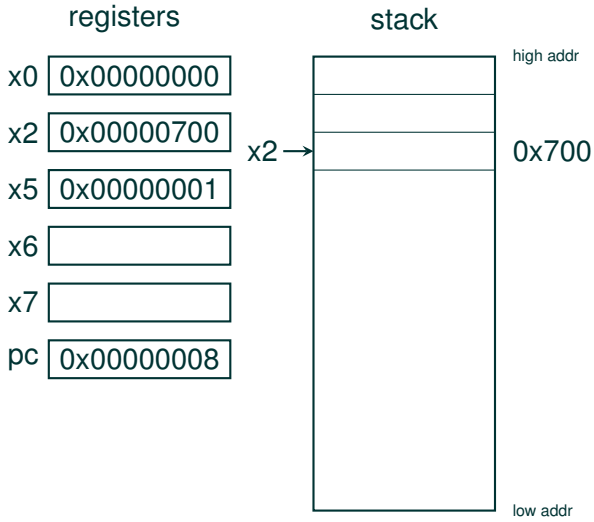
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



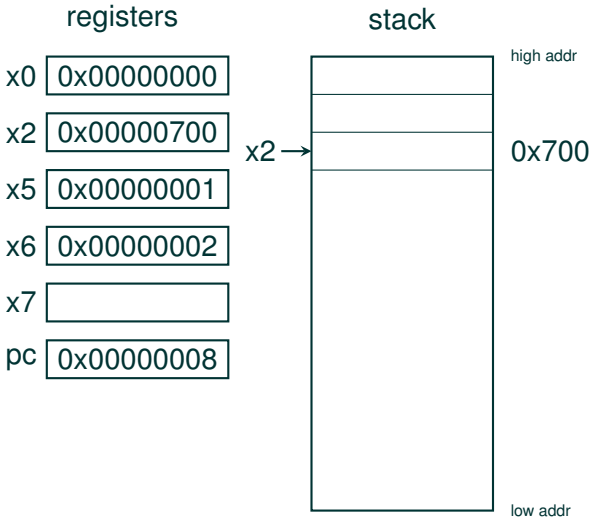
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



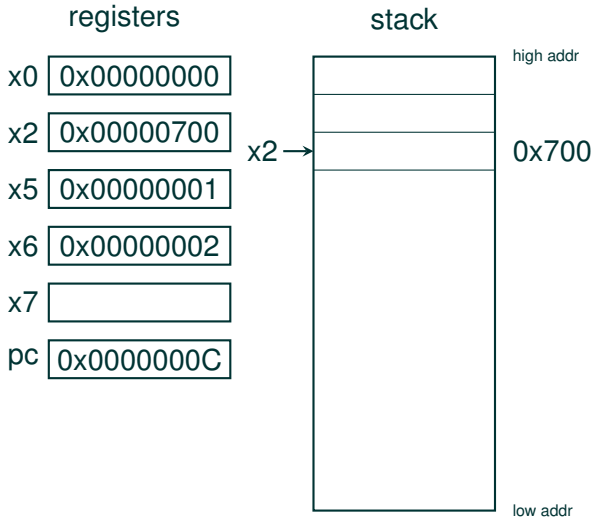
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



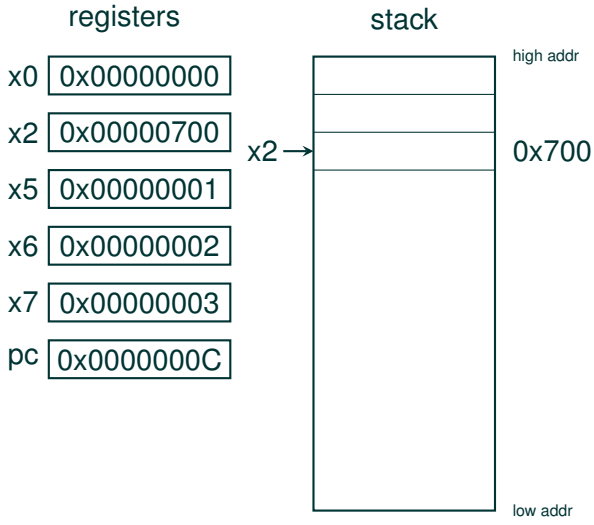
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



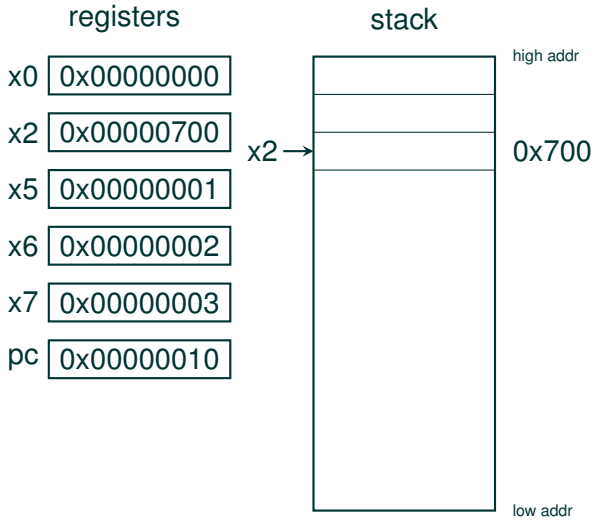
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



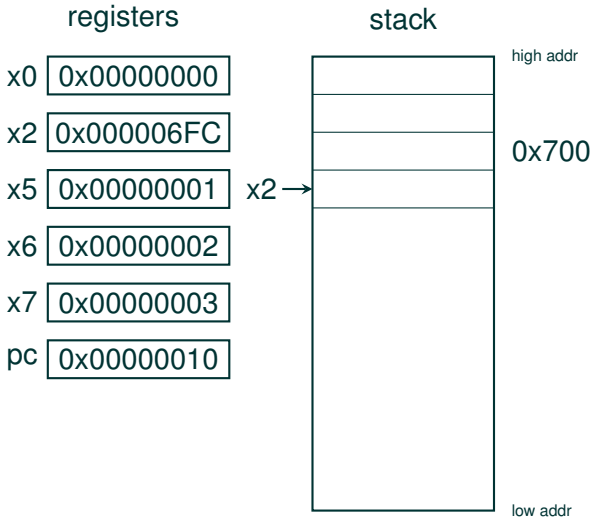
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



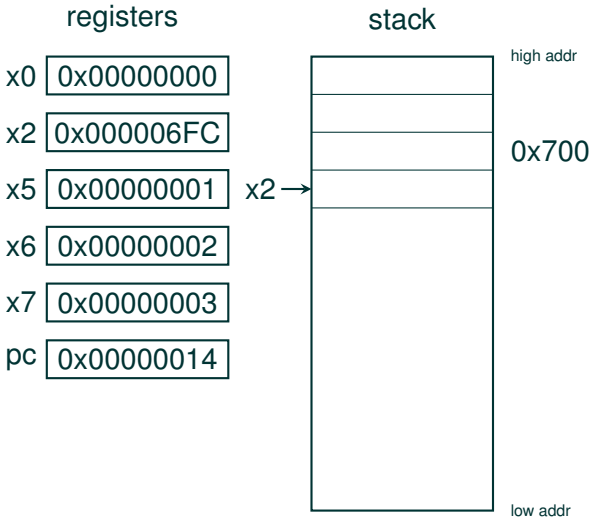
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



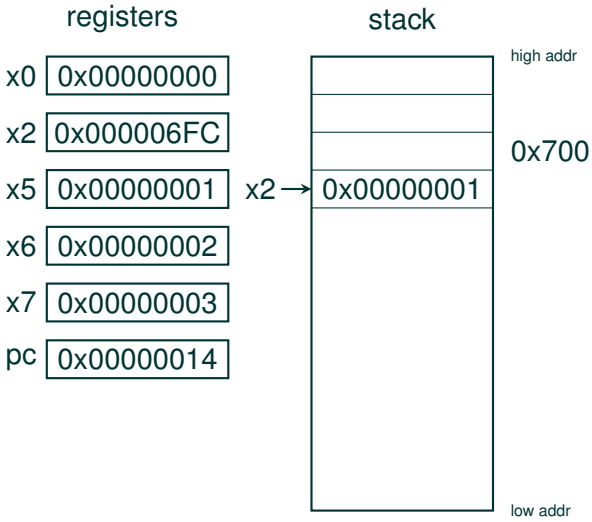
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



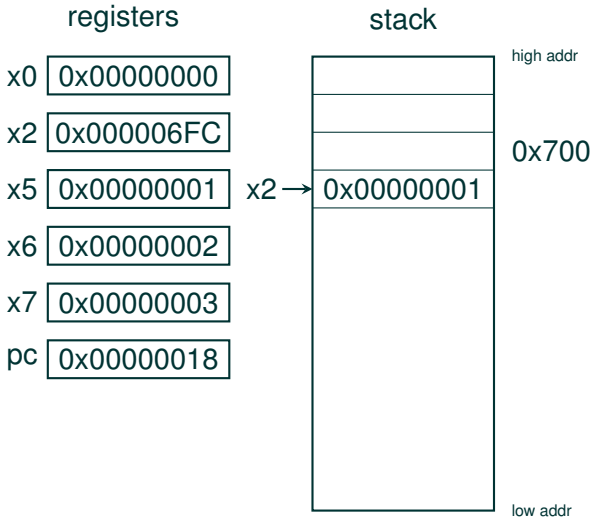
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



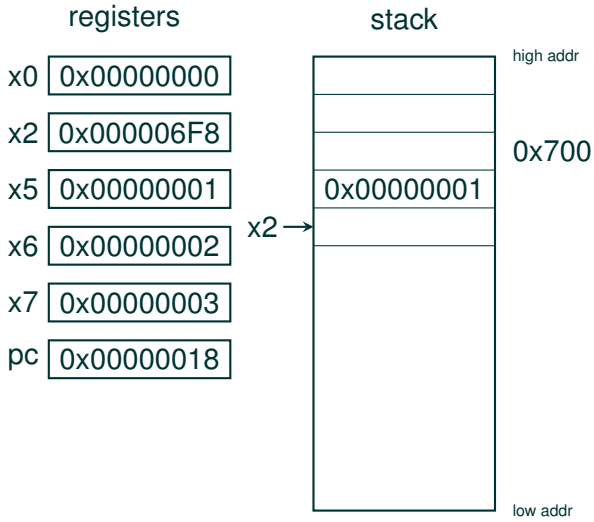
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



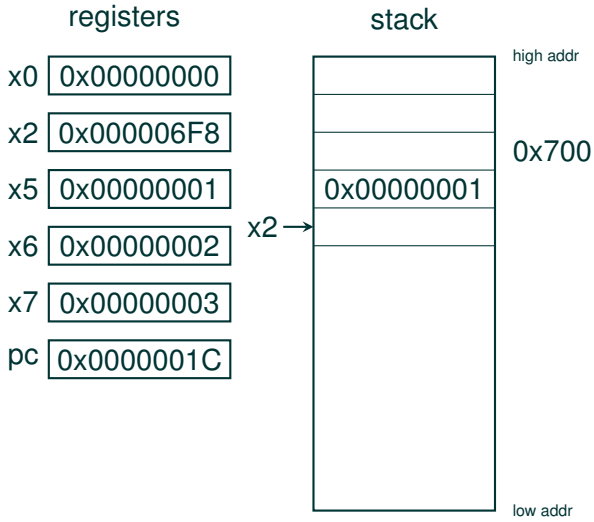
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



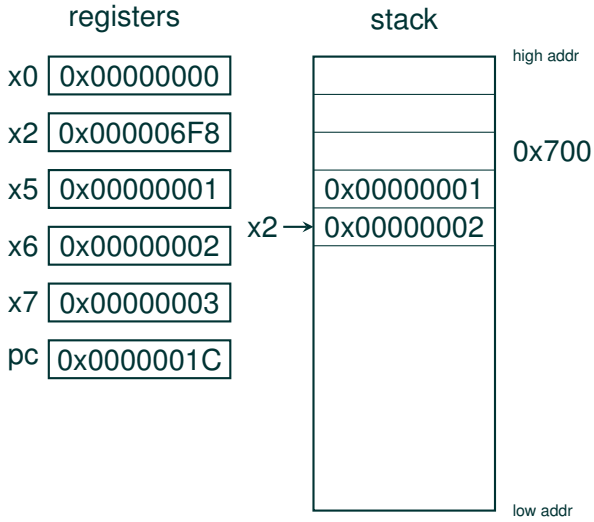
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



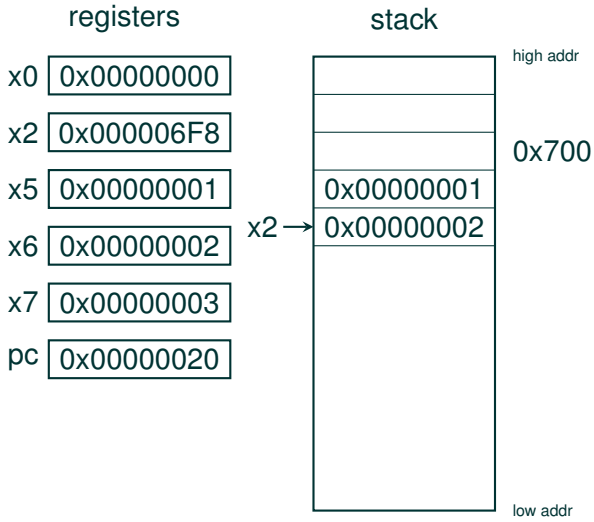
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



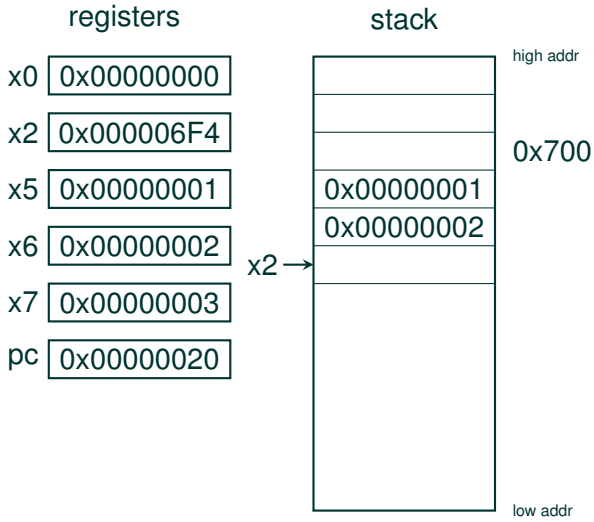
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



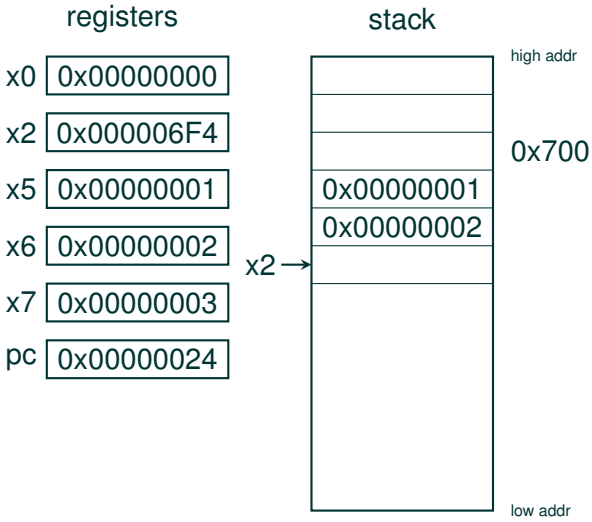
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



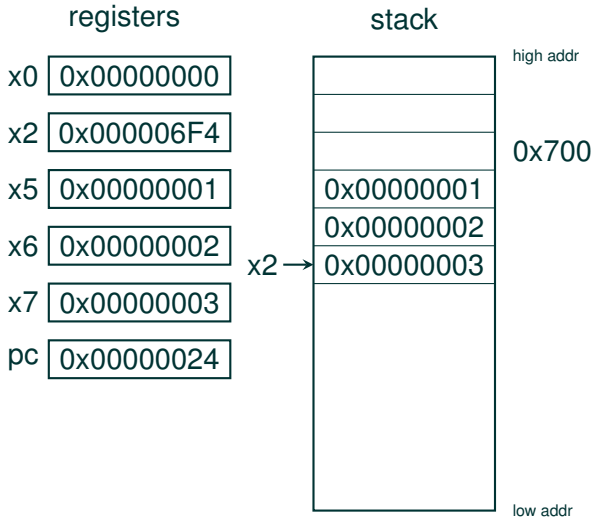
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



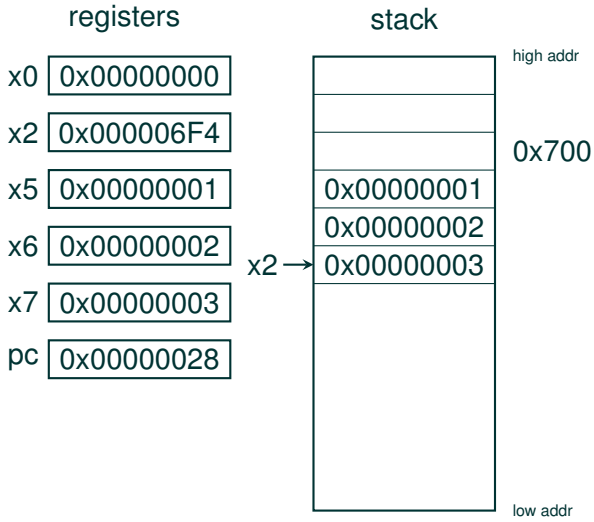
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



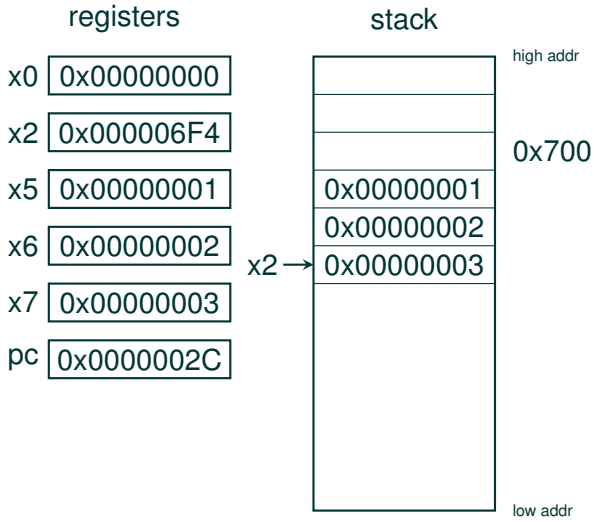
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



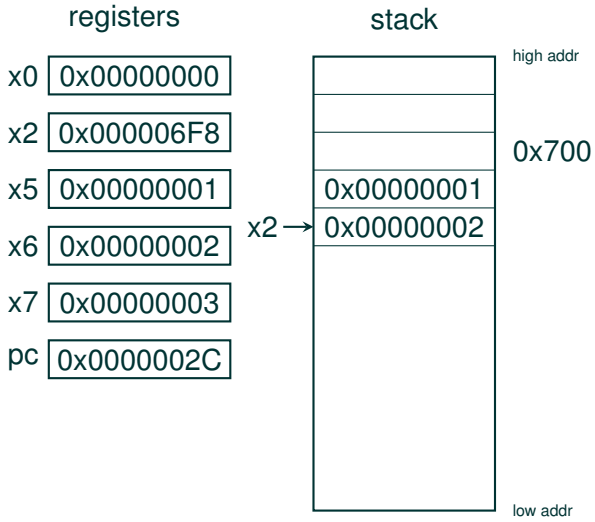
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



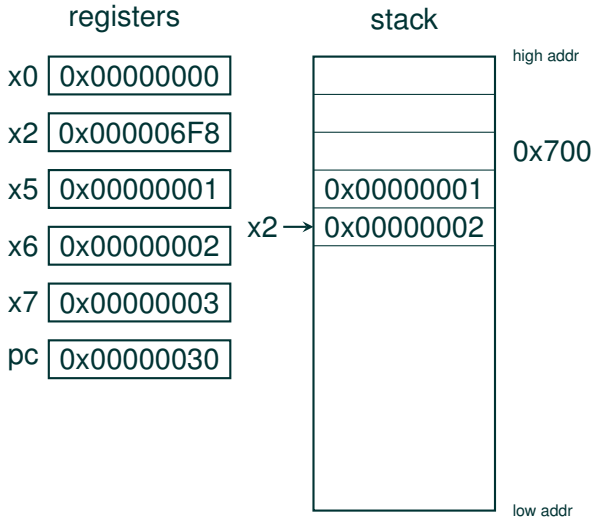
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



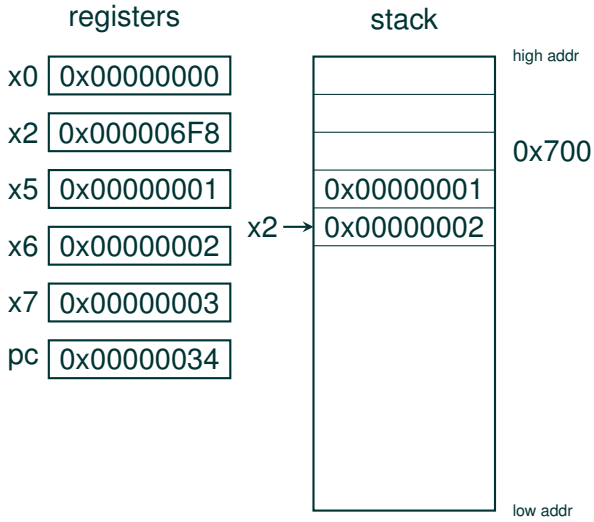
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



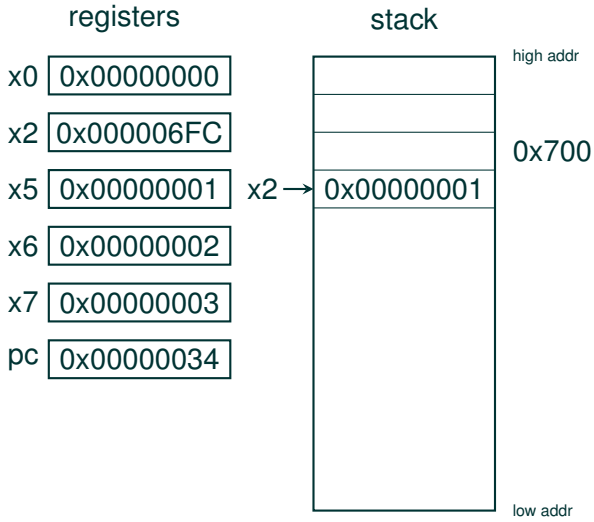
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



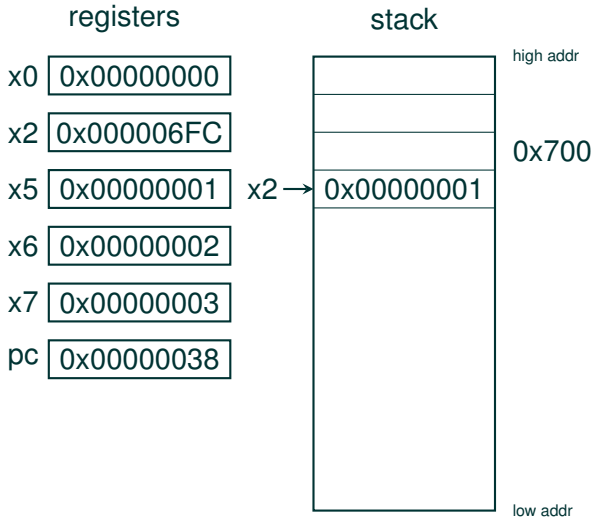
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



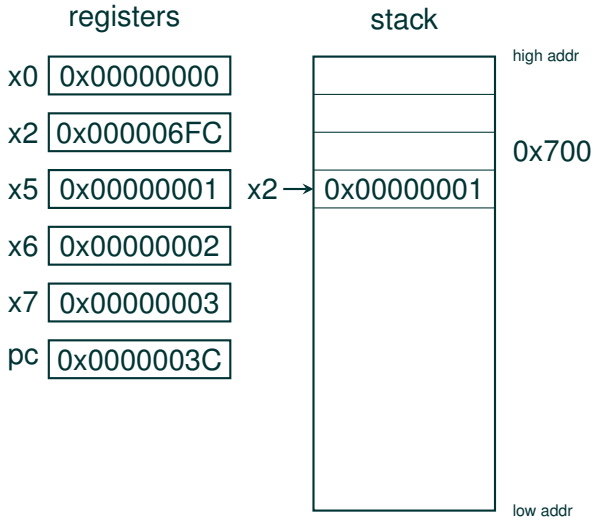
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



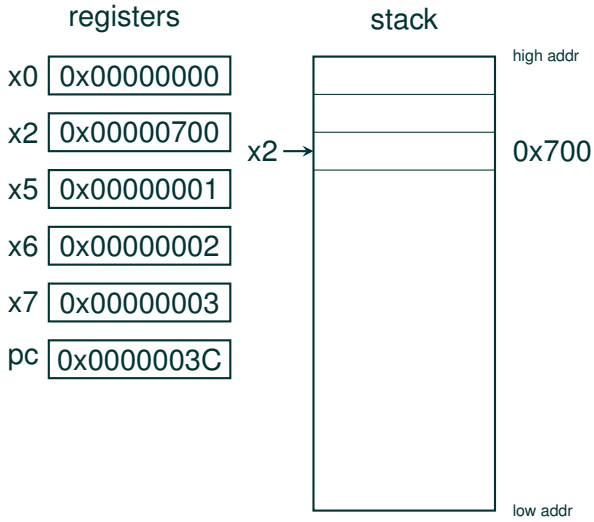
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



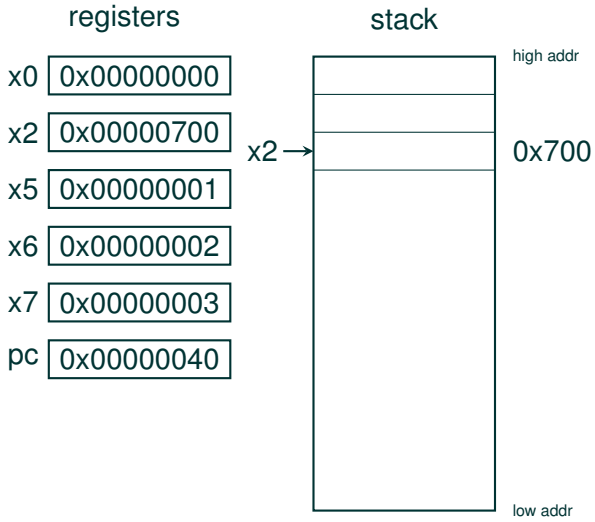
Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```



Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```

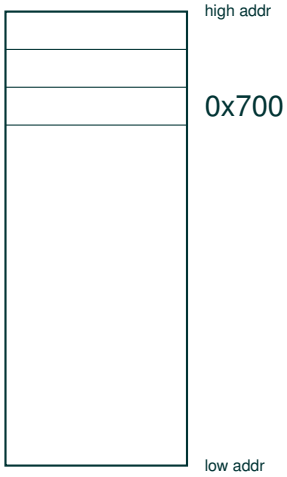


Push and pop

```
ADDI    x2,x0,0x700

ADDI    x5,x0,1
ADDI    x6,x0,2
ADDI    x7,x0,3

ADDI    x2,x2,-4
SW      x5,0(x2)
ADDI    x2,x2,-4
SW      x6,0(x2)
ADDI    x2,x2,-4
SW      x7,0(x2)

LW      x7,0(x2)
ADDI    x2,x2,4
LW      x6,0(x2)
ADDI    x2,x2,4
LW      x5,0(x2)
ADDI    x2,x2,4

EBREAK
```

registers

x0	0x00000000
x2	0x00000700
x5	0x00000001
x6	0x00000002
x7	0x00000003
pc	0x00000040



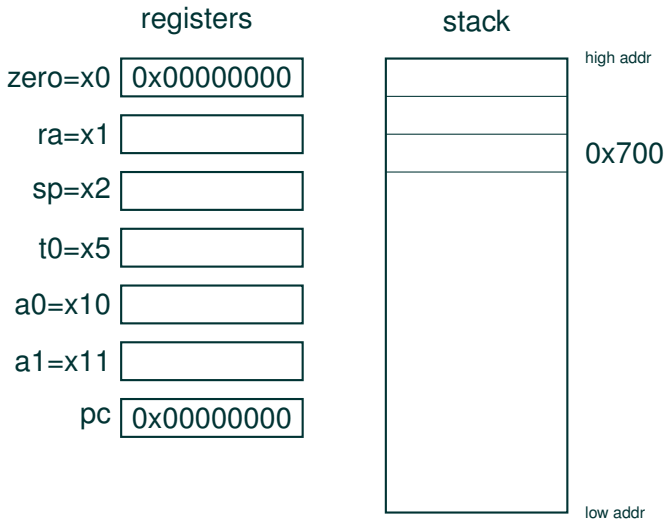
Recursive call on a stack

Recursive call on a stack

```c
// Computes the sum of the arithmetic progression defined by
// result = sum init+(n-1), n=[1,count]
int arith_series(int init, int count) {
    if (1 >= count)
        return init;
    return count + arith_series(init, count-1);
}

int main(void) {
    return arith_series(1, 3);
}
```

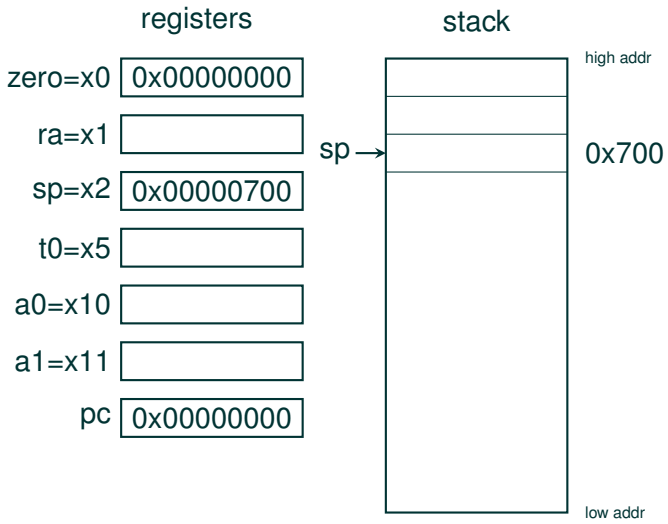
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



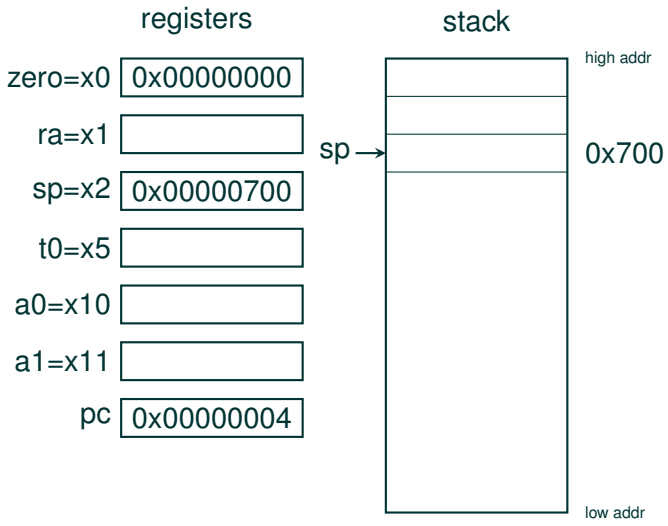
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



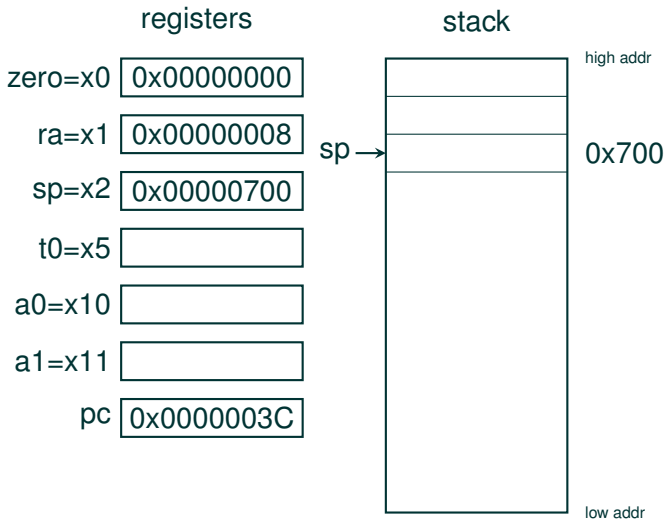
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



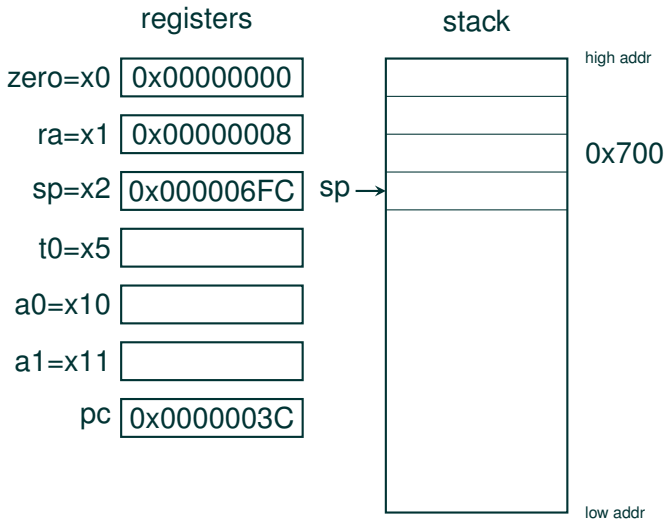
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



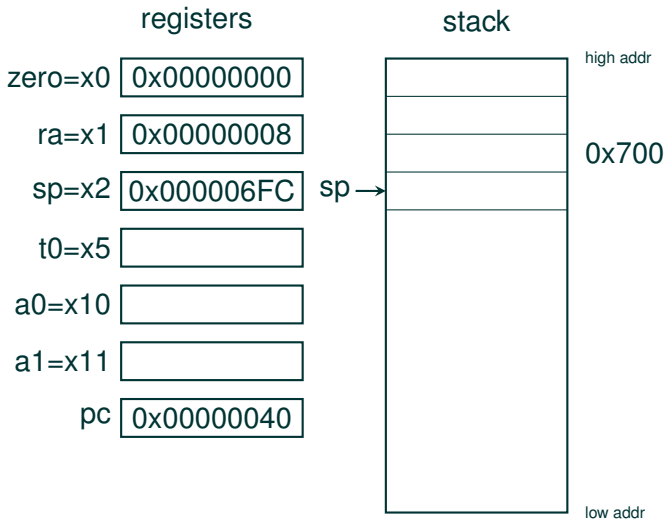
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



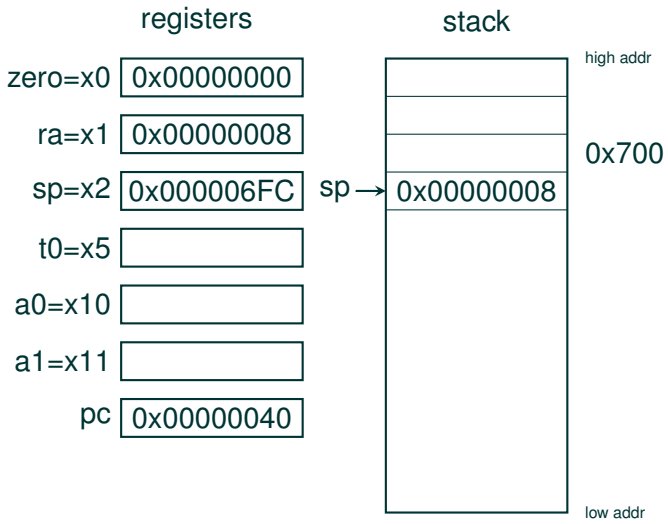
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



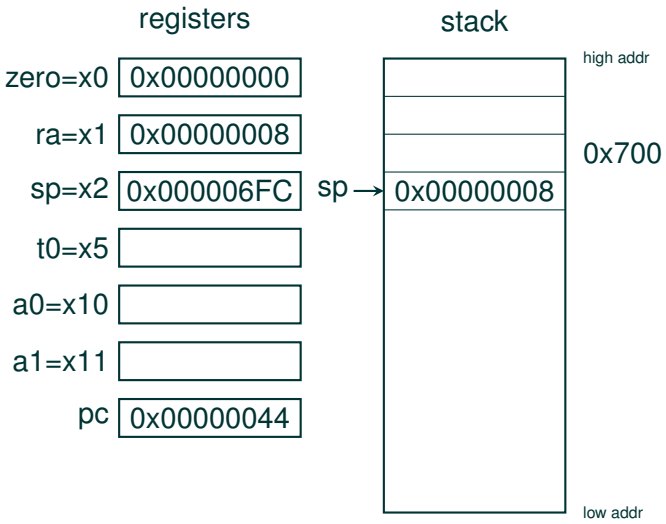
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



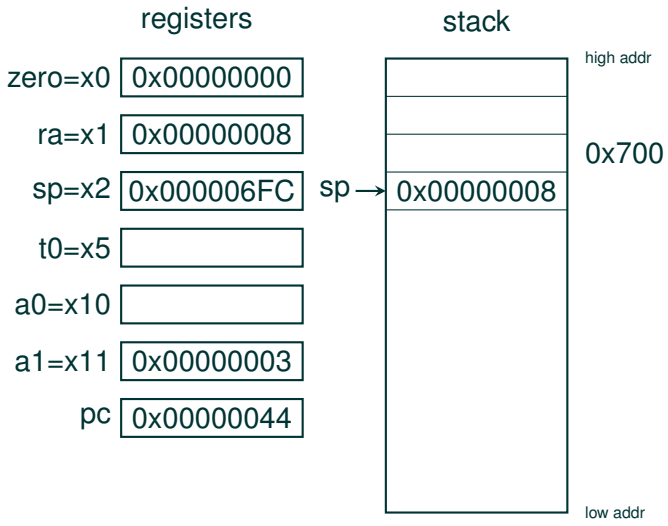
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



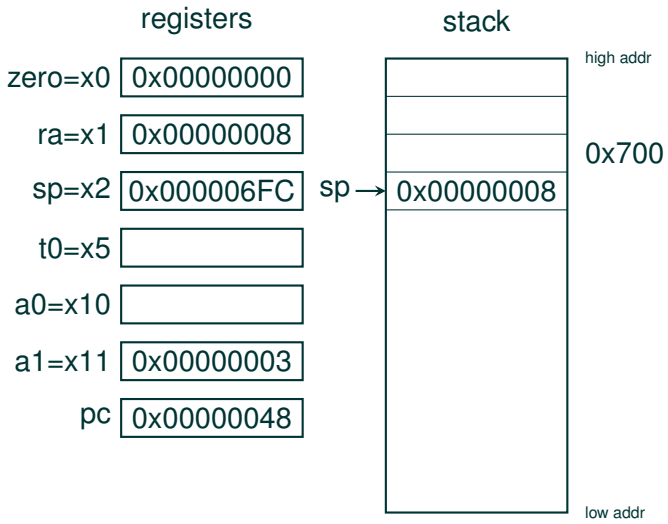
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



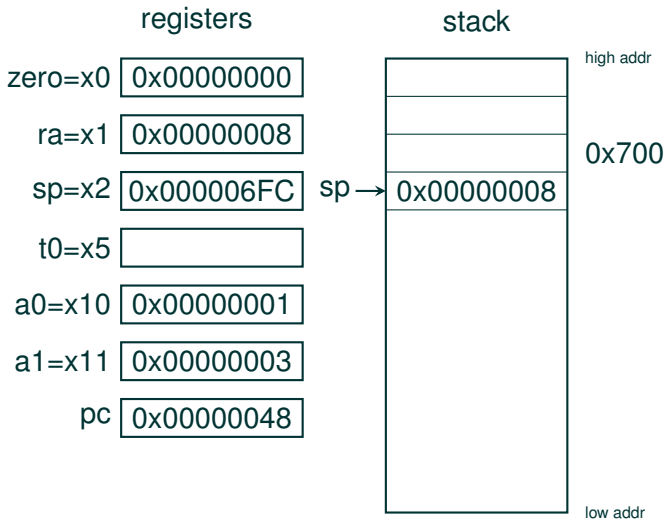
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



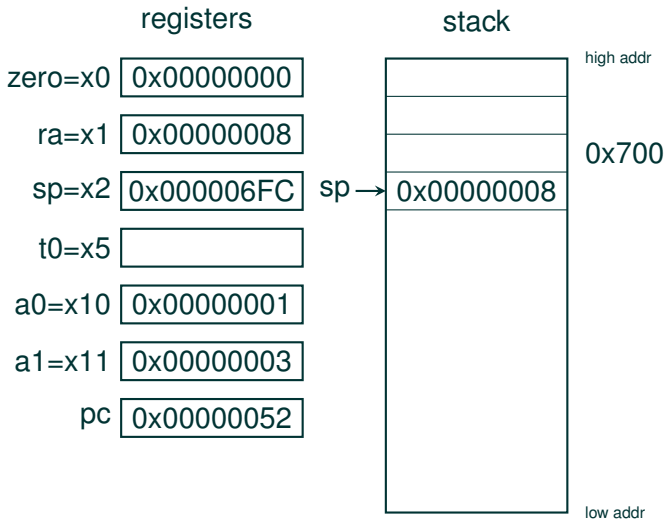
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



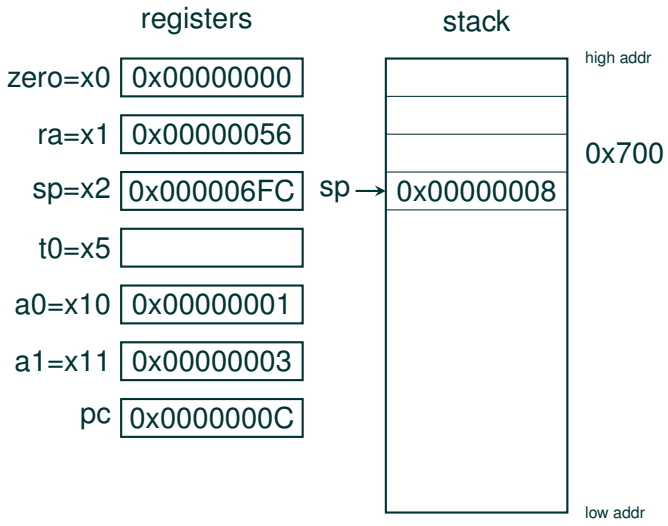
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



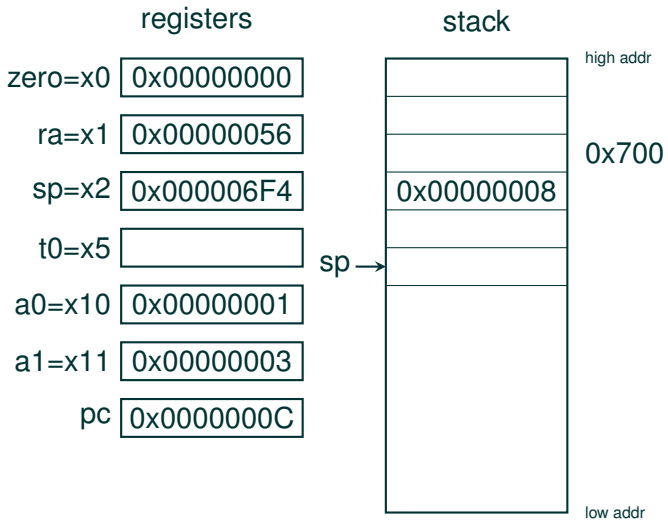
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



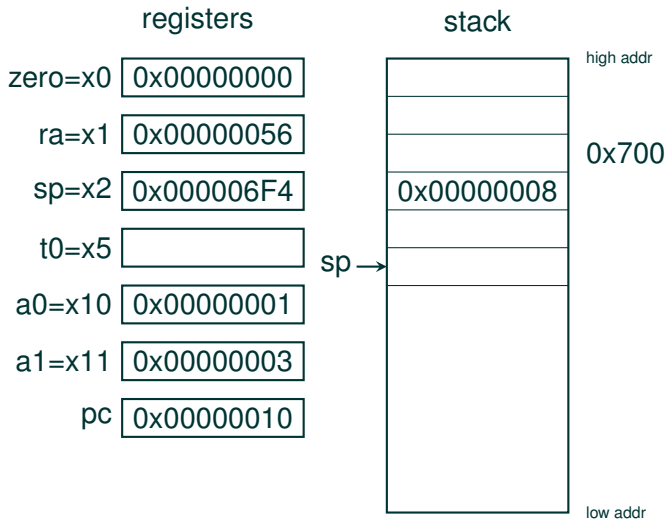
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



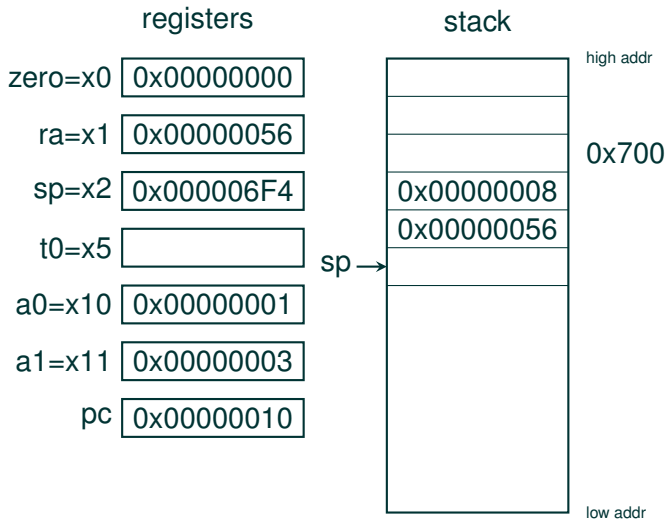
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



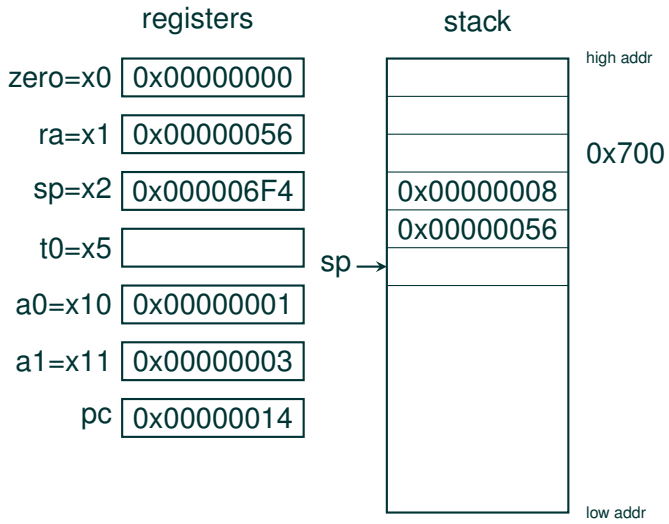
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



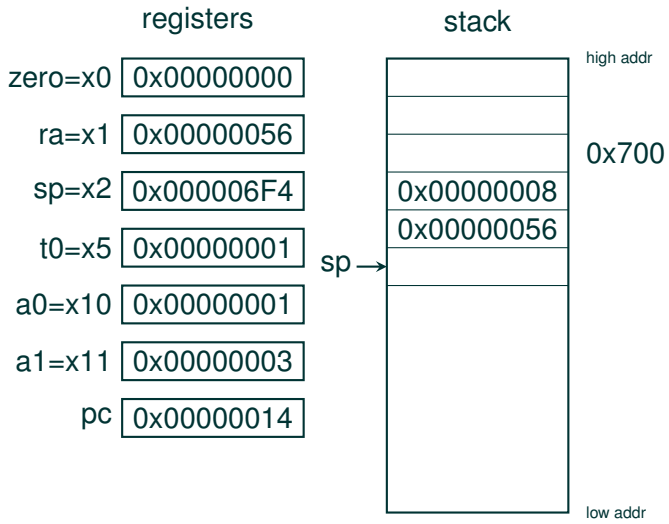
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



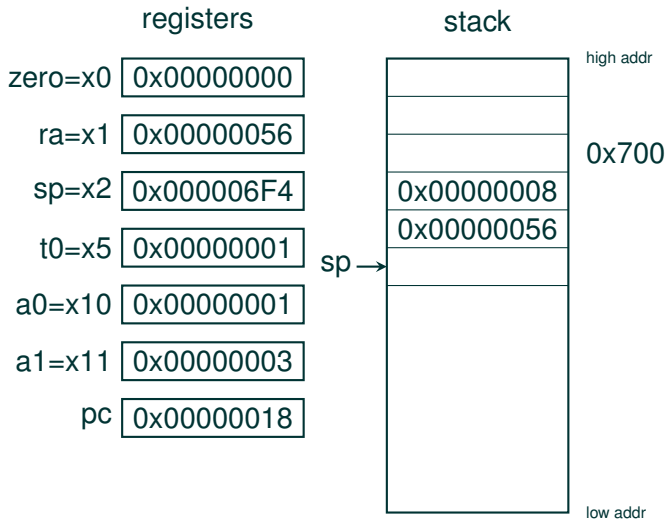
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



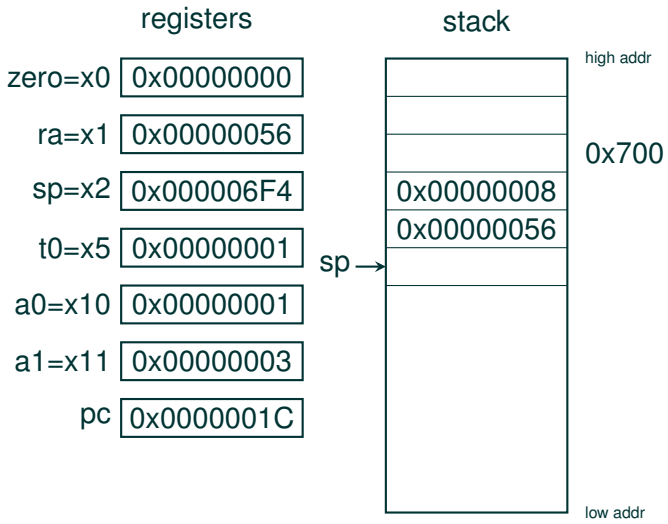
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



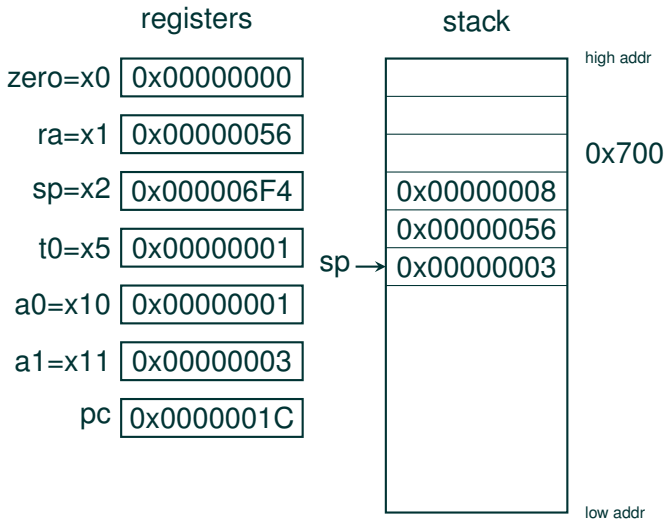
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



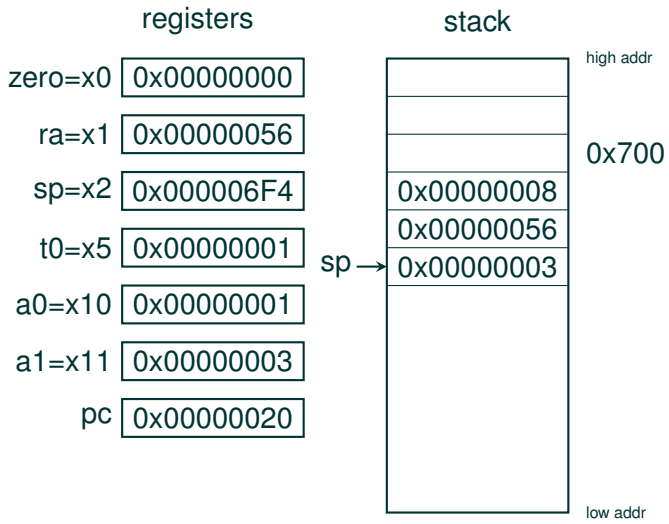
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



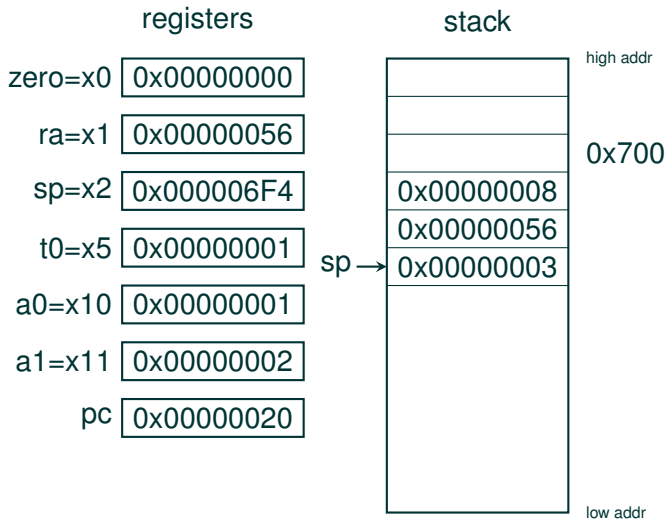
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



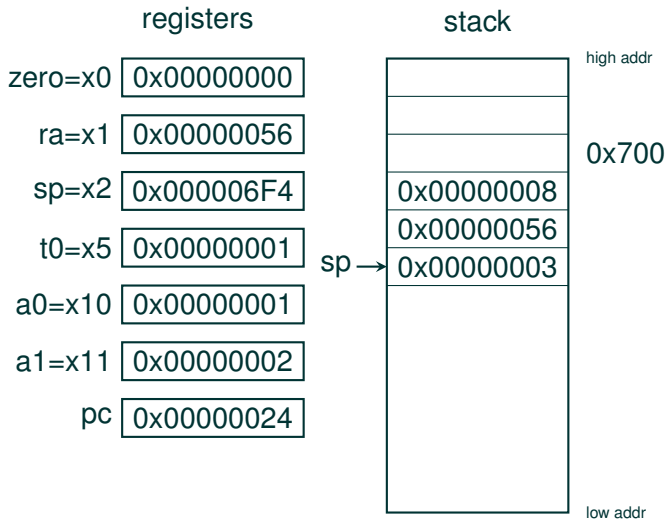
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



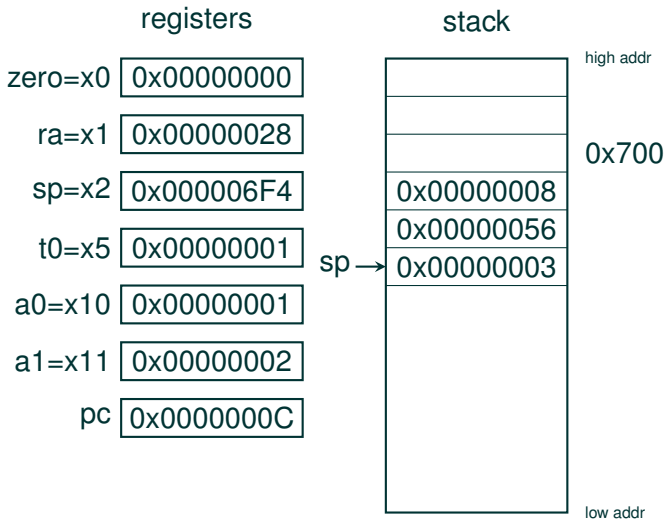
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



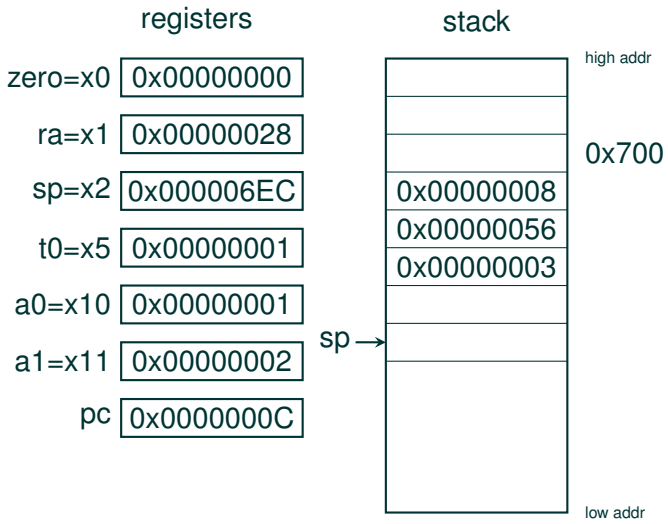
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



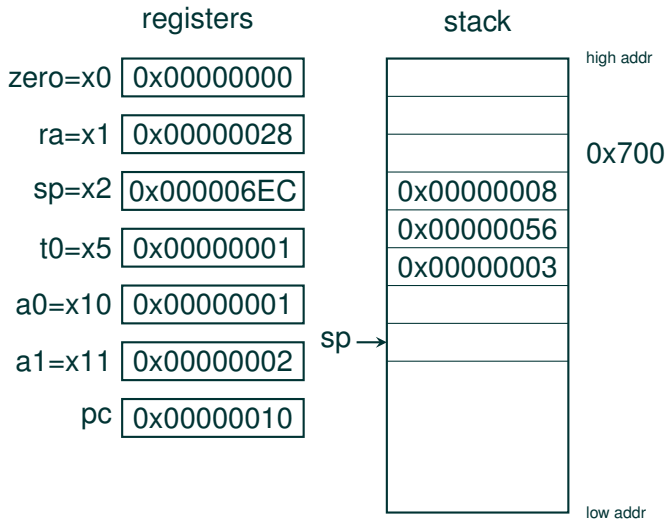
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



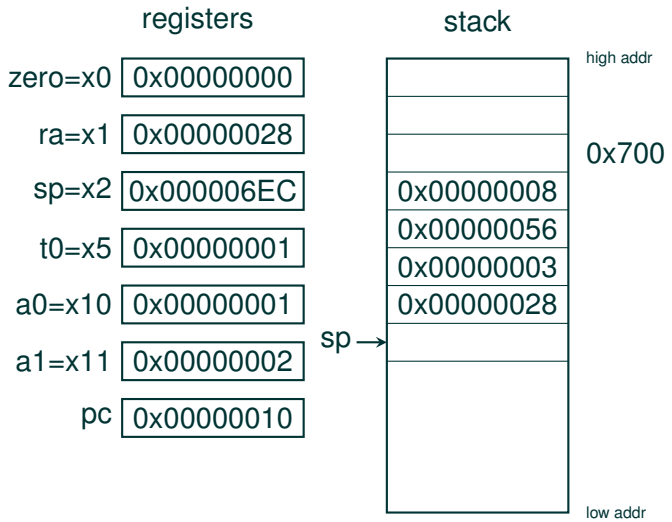
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



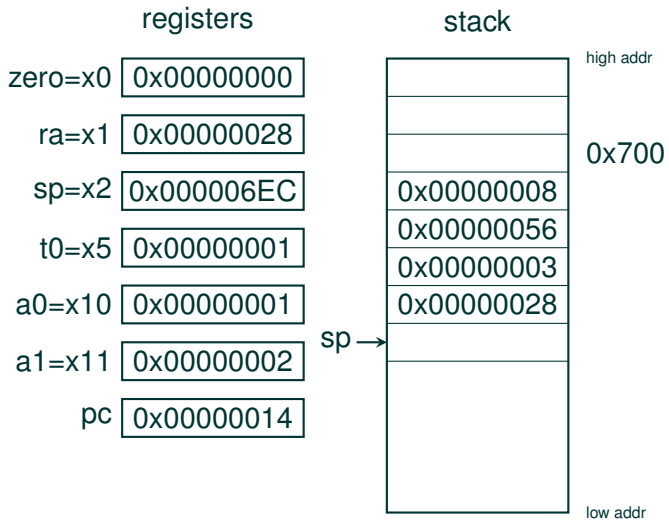
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



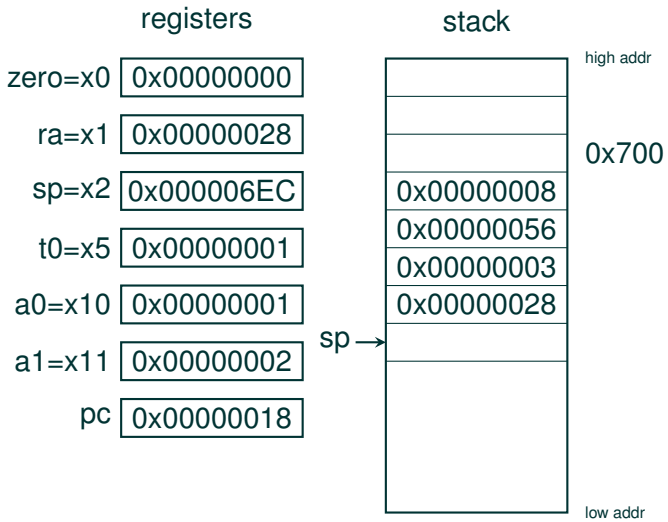
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



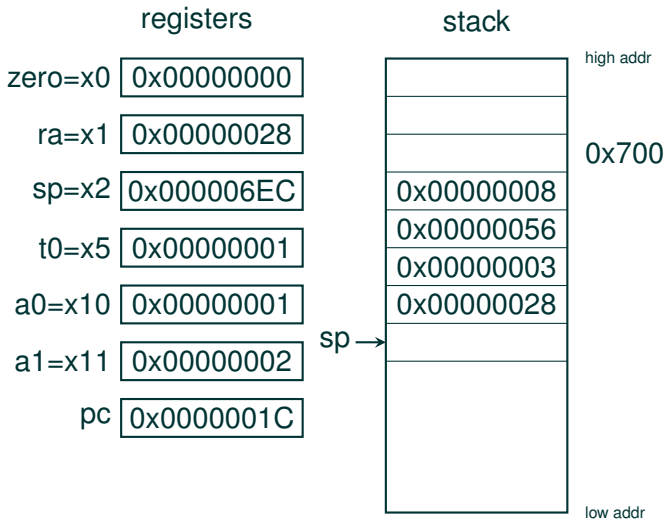
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



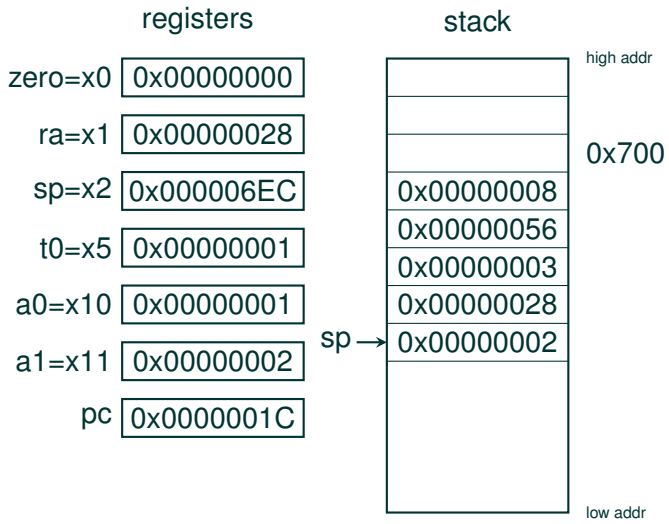
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



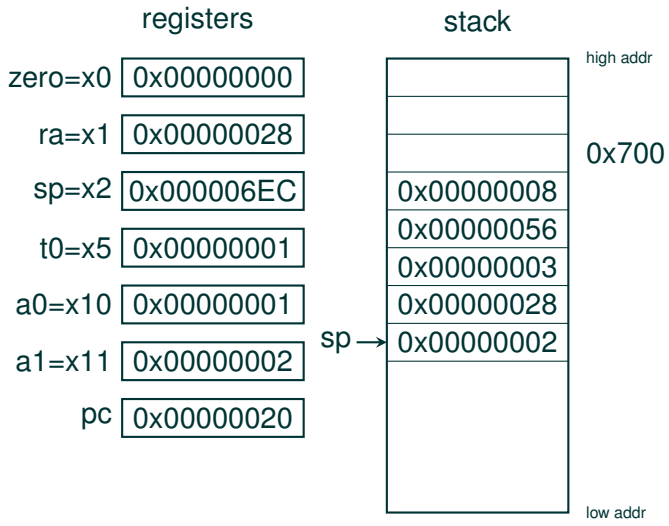
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



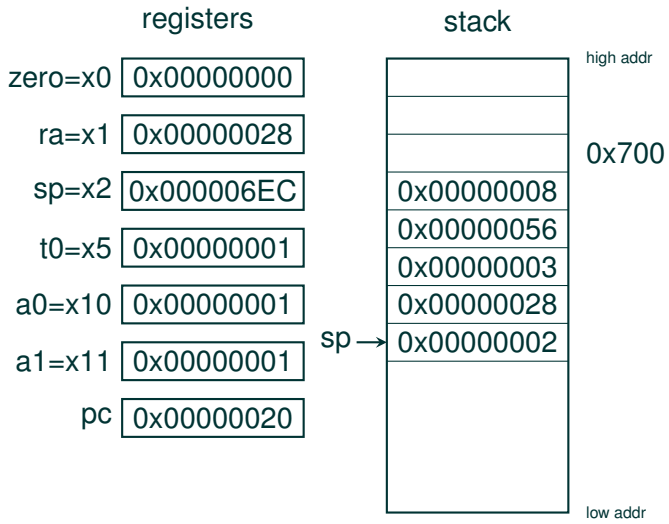
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



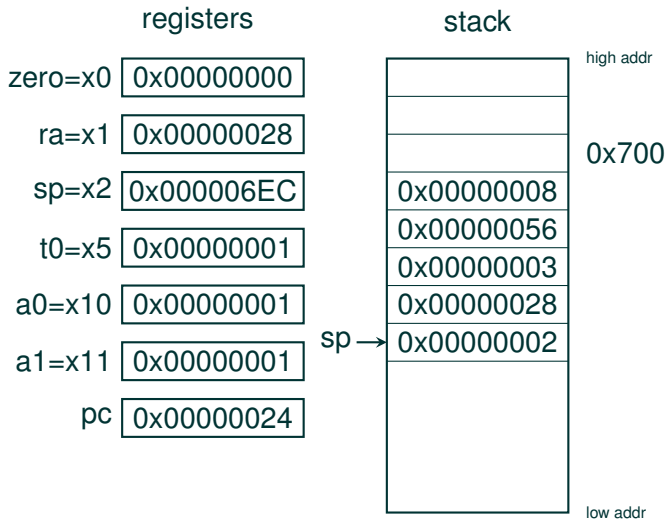
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



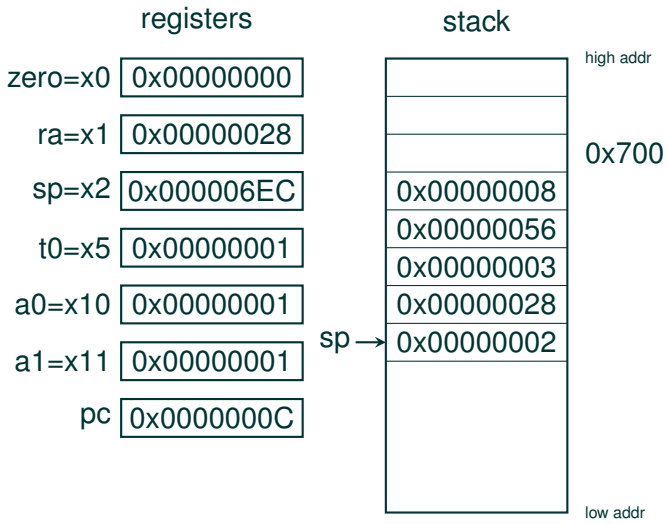
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



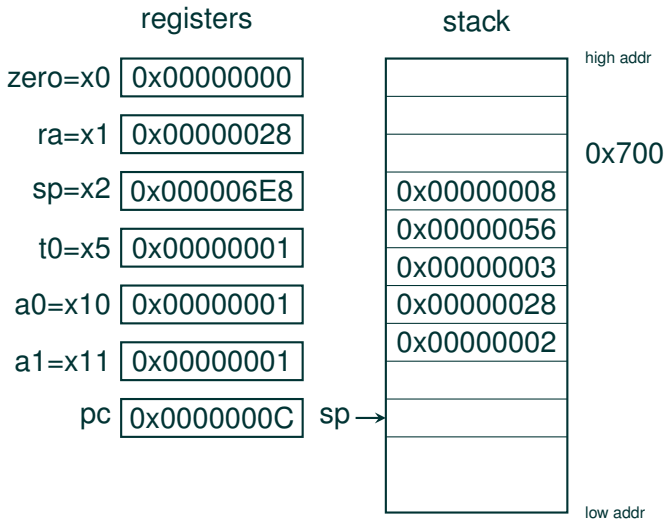
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



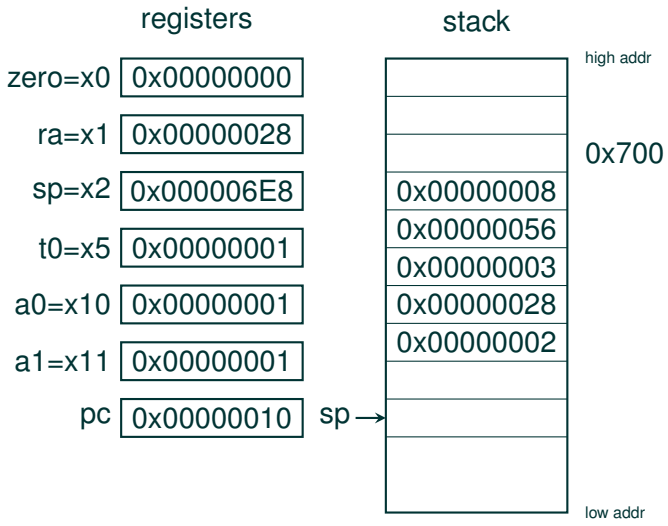
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



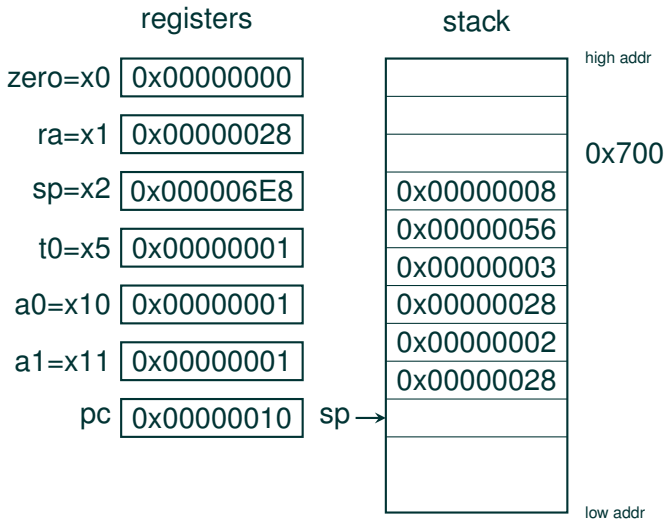
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



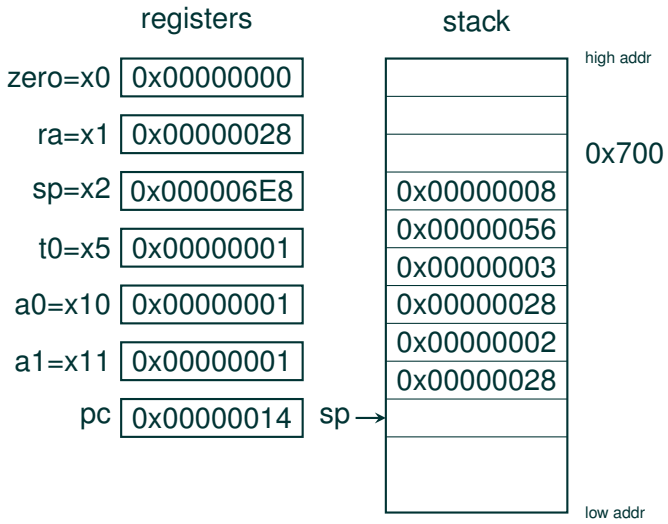
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



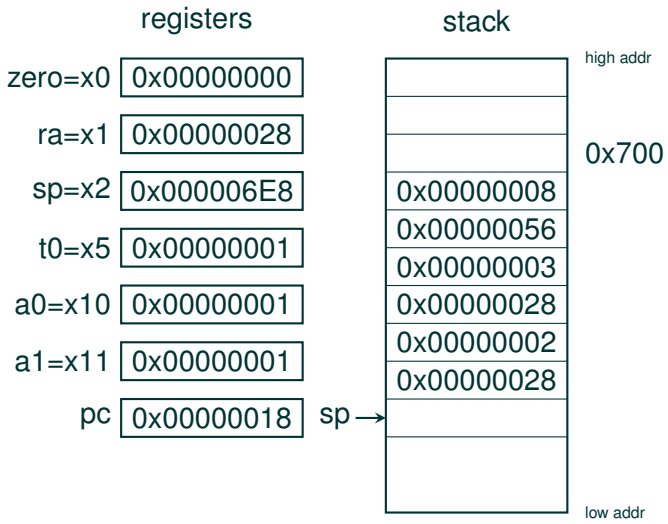
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



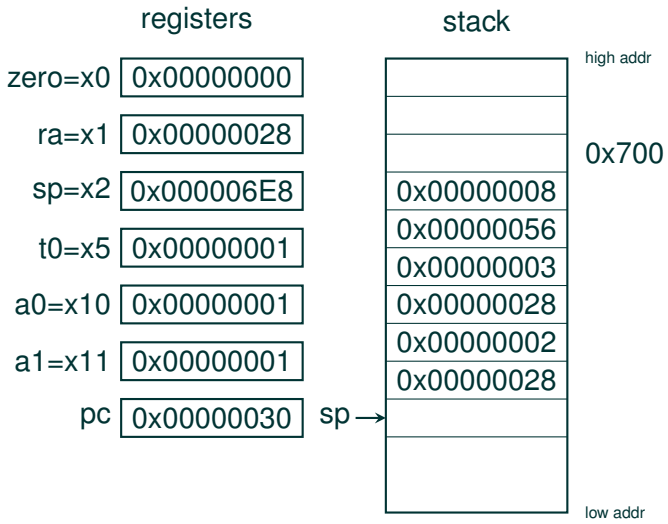
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



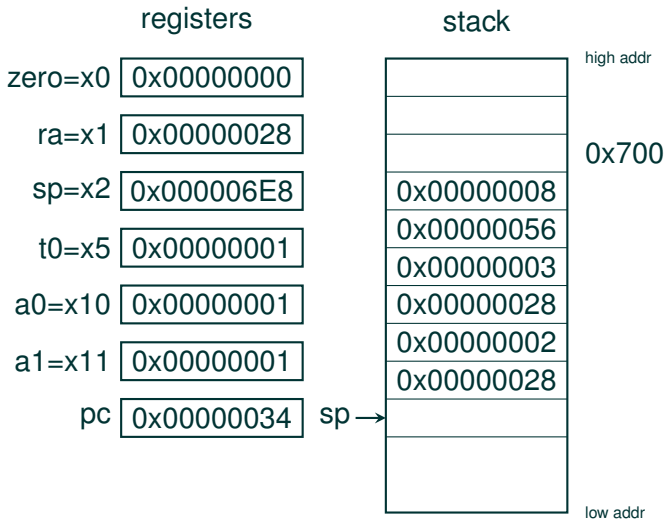
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



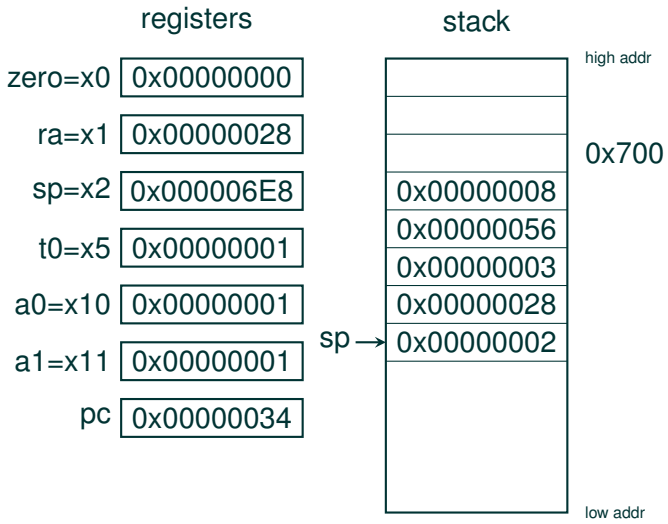
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



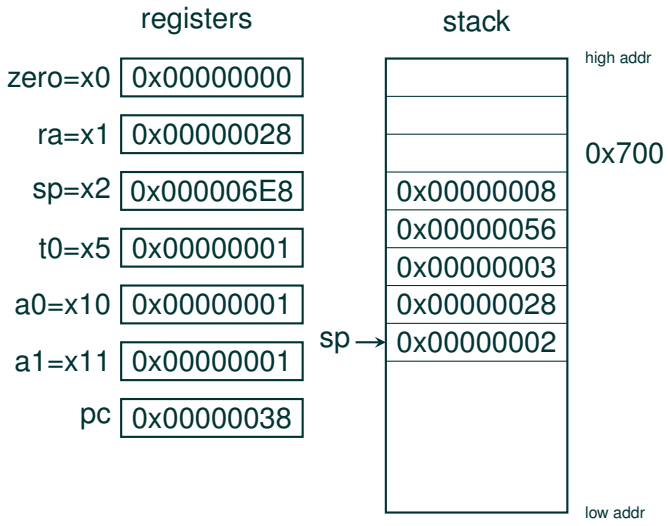
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



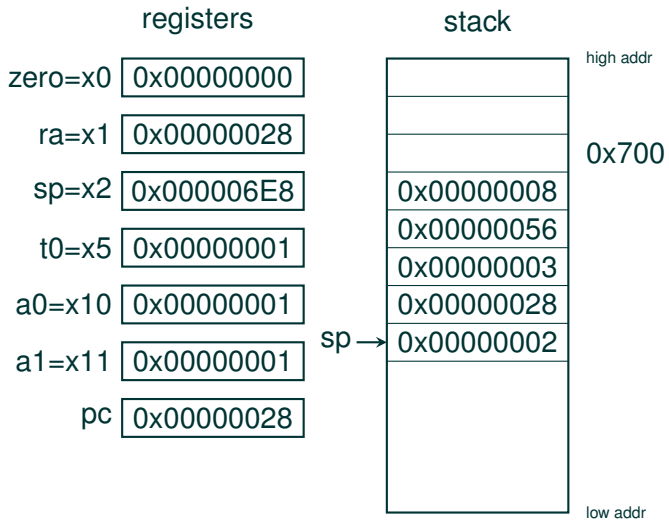
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



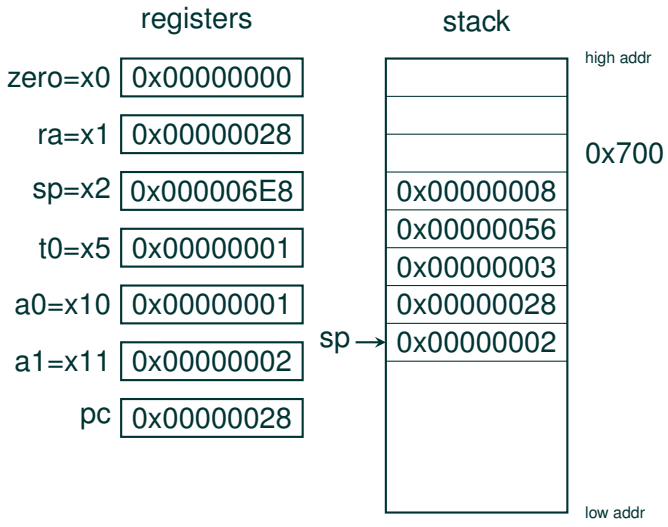
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



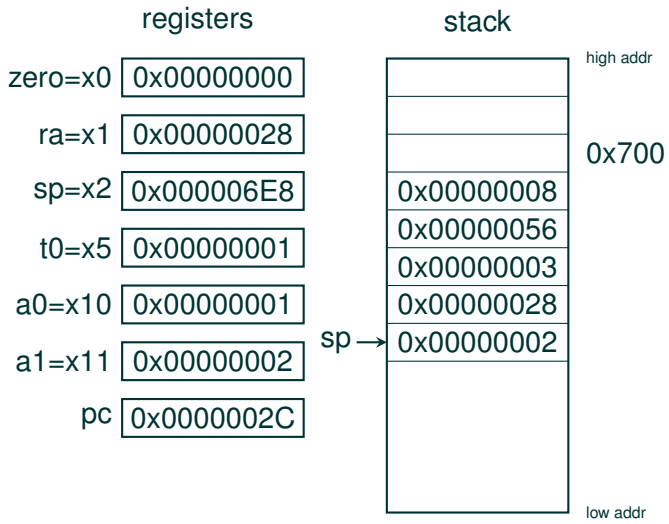
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



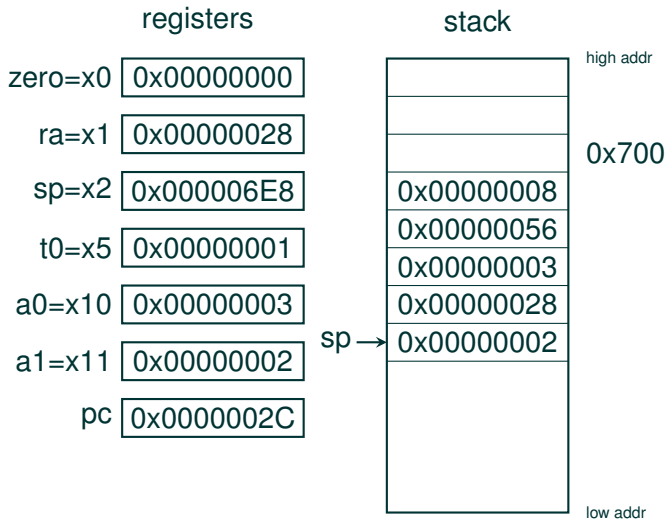
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



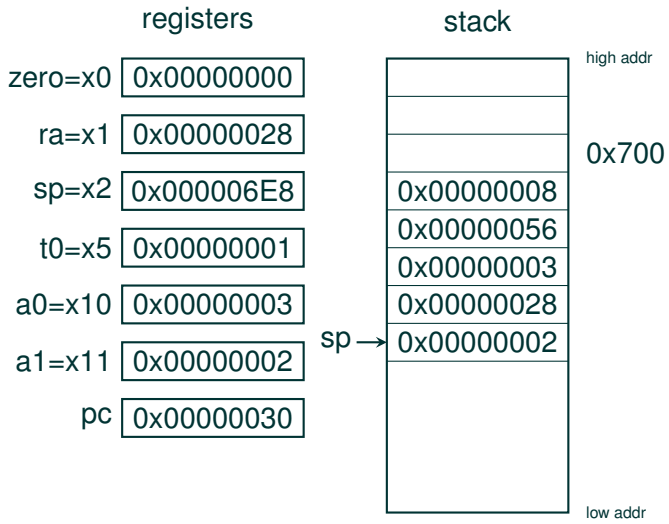
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



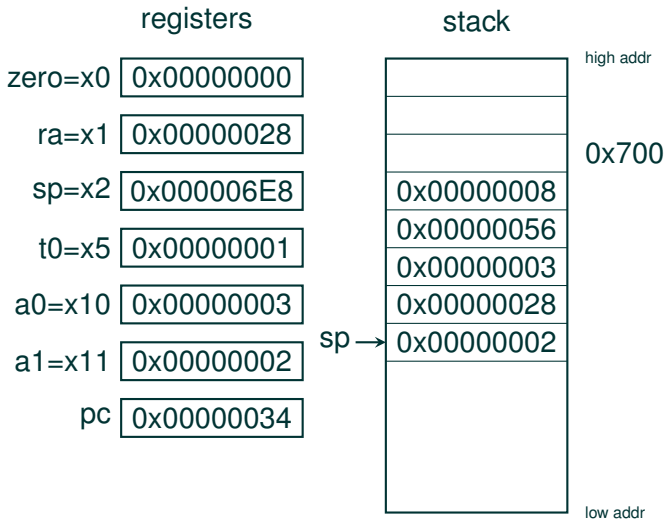
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



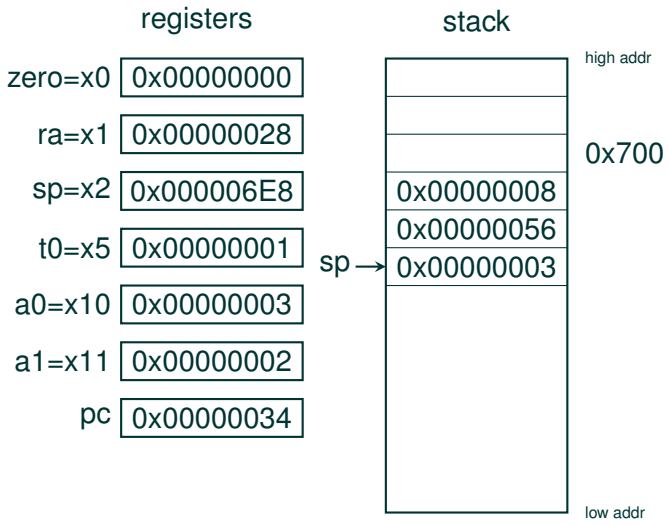
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



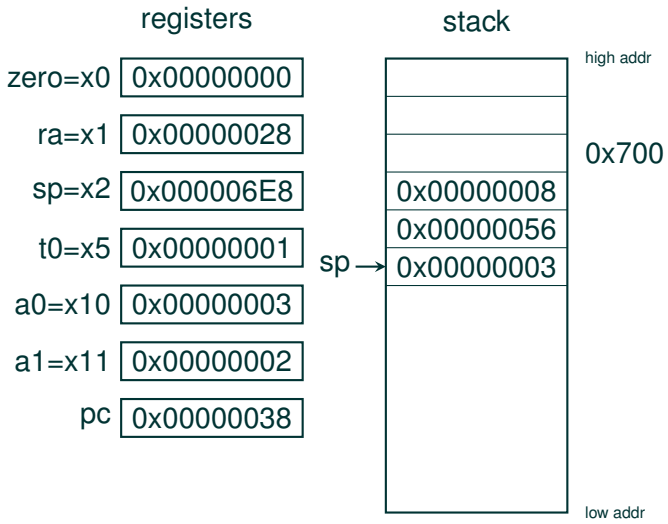
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



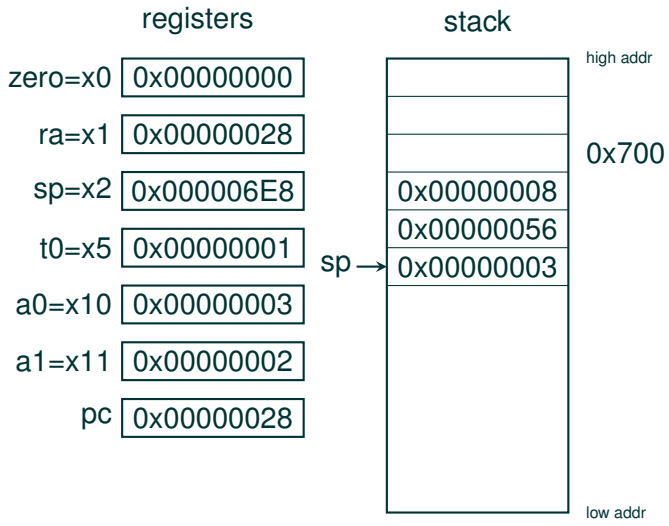
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



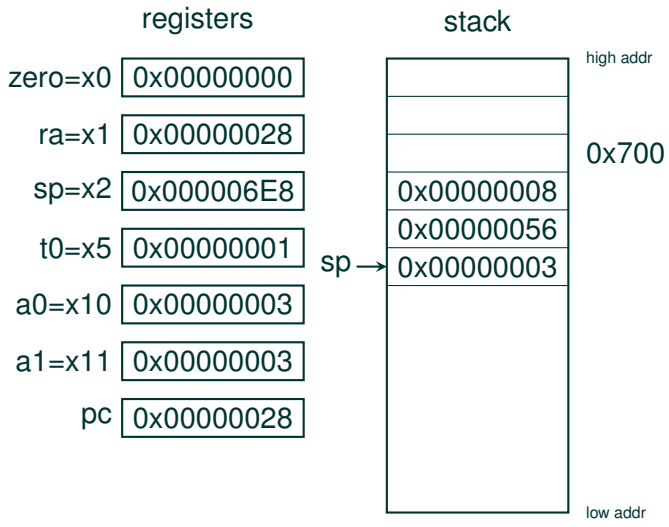
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



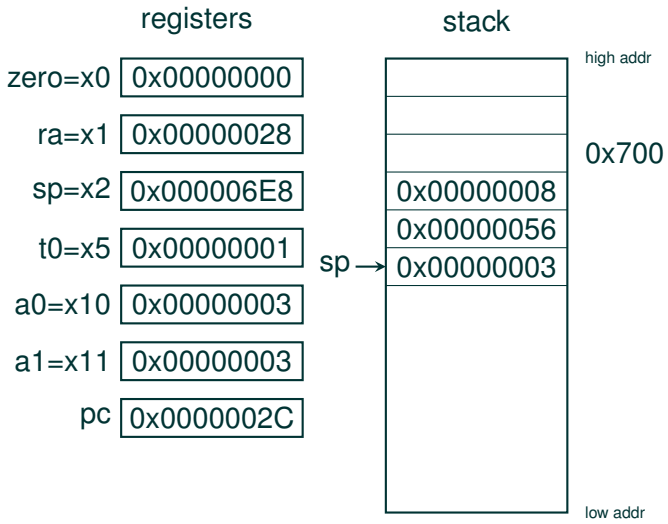
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



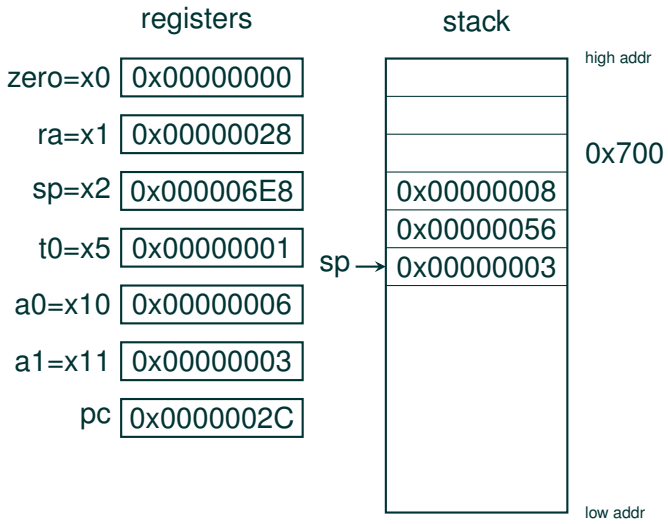
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



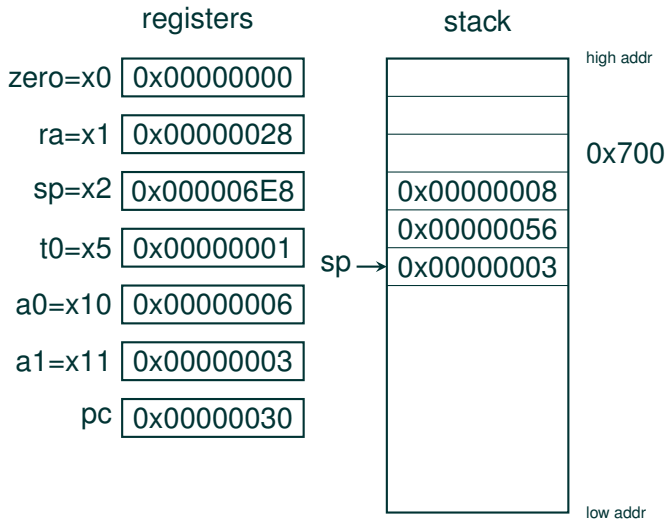
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



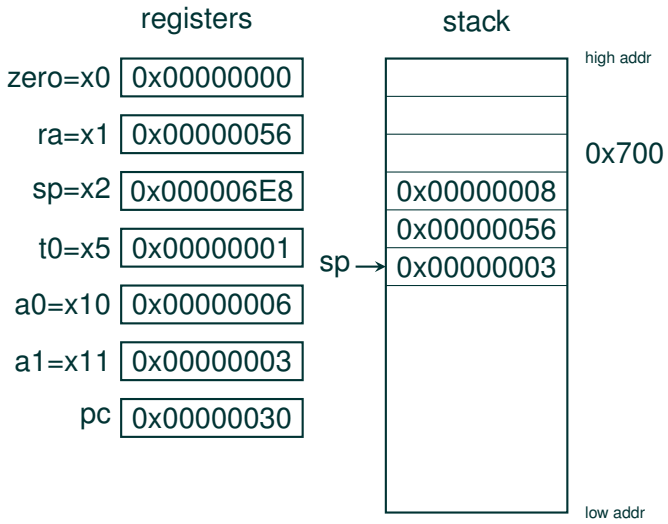
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



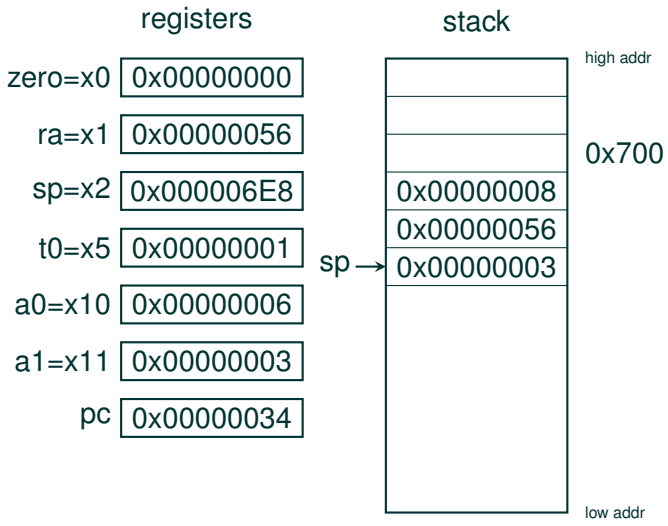
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



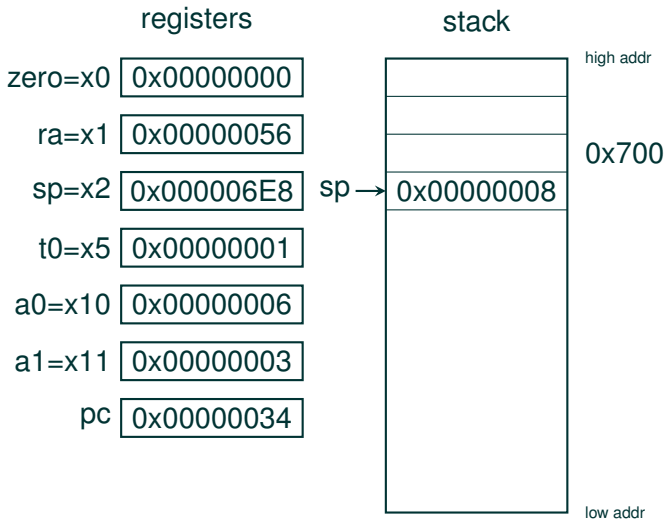
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



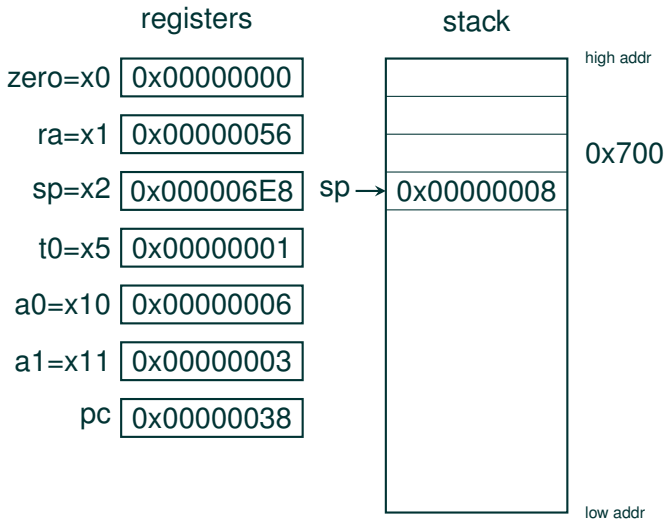
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



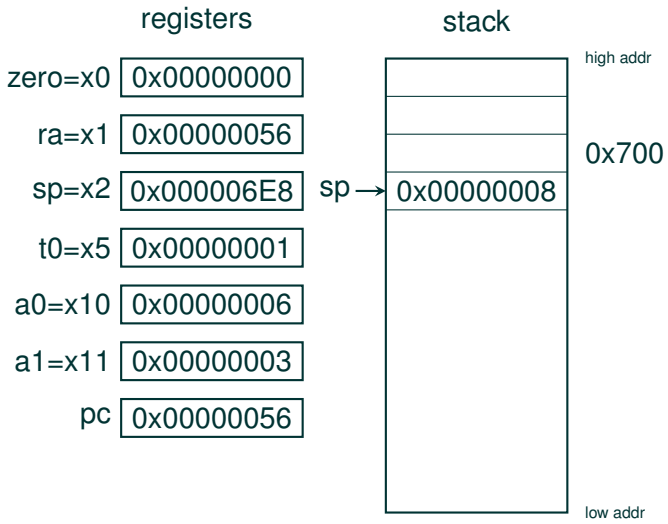
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



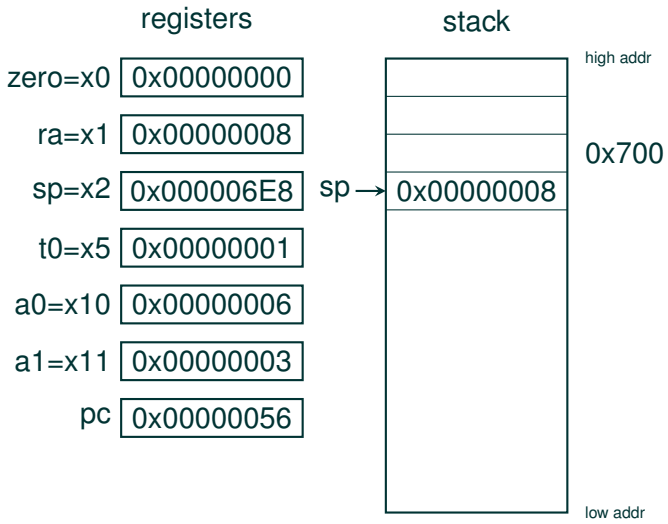
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



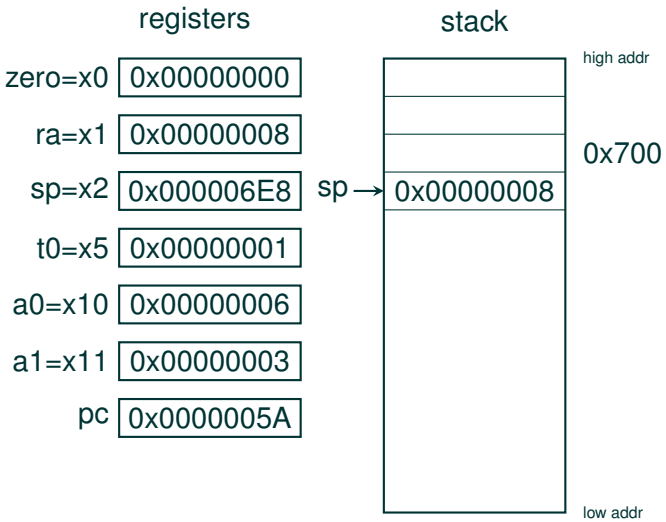
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



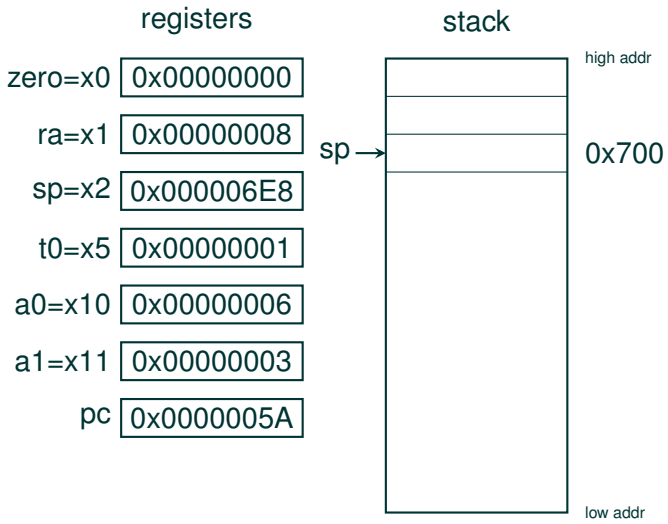
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



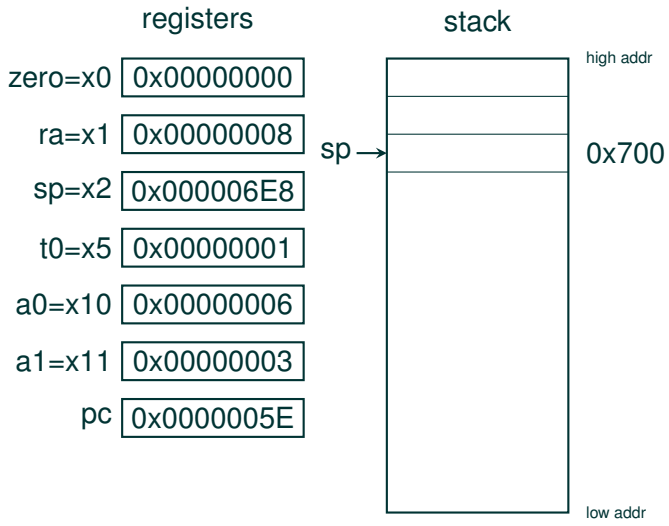
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



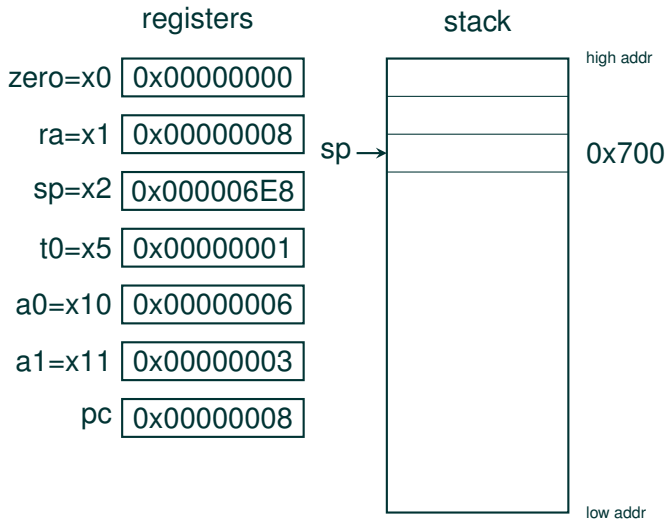
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



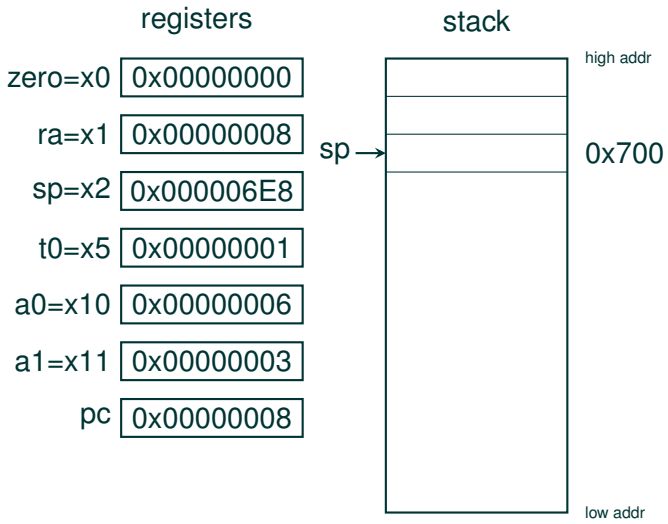
Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



Recursive call on a stack

```
_start:
    ADDI sp, zero, 0x700
    JAL ra, main
    EBREAK
arith_series:
    ADDI sp, sp, -8
    SW ra, 4(sp)
    ADDI t0, zero, 1
    BGE t0, a1, arith_series_return
    SW a1, 0(sp)
    ADDI a1, a1, -1
    JAL ra, arith_series
    LW a1, 0(sp)
    ADD a0, a0, a1
arith_series_return:
    LW ra, 4(sp)
    ADDI sp, sp, 8
    JALR zero, 0(ra)
main:
    ADDI sp, sp, -4
    SW ra, 0(sp)
    ADDI a1, zero, 3
    ADDI a0, zero, 1
    JAL ra, arith_series
    LW ra, 0(sp)
    ADDI sp, sp, 4
    JALR zero, 0(ra)
```



Function Calls & Stack examples

Stefan Mangard

Computer Organization and Networks
Graz University of Technology